EECS 330 Lab 05: Hash Table with Separate Chaining

Objective

- Get familiar with the implementation of hash table with separate chaining in C++.
- Get familiar with dynamic hash table size adjustment and rehashing operations.
- Get familiar with universal hash function calculations and modulo speed up with Mersenne prime.

Specifications

1. The hash table should implement the separate chaining approach to handle collisions.
2. The primary hash table should be implemented using the `MyVector` class, and the secondary chain for each bucket should be implemented using the `MyLinkedList` class.
3. The hash table should accept either `std::string` or `long long` as the key data type, and it should support any value data type.
4. The hash table should be able to dynamically adjust its table size using table doubling and halving.
5. The hash table should support data retrieval, insertion, and deletion in constant time on average.

Hints

1. You can make use of the `HashFunc` and `HashedObj` classes that have already been implemented.
2. You can use Sieve of Eratosthenes to precompute prime numbers.
3. Recall the difference between `MyVector::size()` and `MyVector::capacity()`, make the proper use of them.

Testing and Grading

We will test your implementation using the tester main function posted online. The posted input and output examples should be used for a testing purpose, while we will use another set of inputs for grading. Your code will be compiled under Ubuntu 20.04 LTS using g++ version 9.3.0 (default) with C++11 standard.

Your final score will be the determined by the success percentage of your program when fed with many random inputs. Note that if your code does not compile (together with our tester main function), you will receive 0. Therefore, it is very important that you ensure your implementation can be successfully compiled before submission.

Submission and Deadline

Please submit your implementation as three .h files, with names "MyVector_[YourKUID].h" (the same as your Lab 01 submission), "MyLinkedList_[YourKUID].h" (the same as your Lab 02 submission), and "MyHashTable_[YourKUID].h". For example, if my KU ID is c123z456, my submission will be three files named "MyVector_c124z456.h", "MyLinkedList_c124z456.h", and "MyHashTable_c124z456.h". Please zip all files into one single file (with the ".zip" extension) before uploading to Canvas. Submissions that do not comply with the naming specification will not be graded. The deadline is Friday Apr 7th, 2023, 11:59PM.